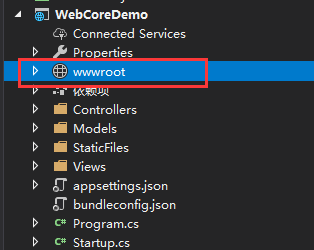
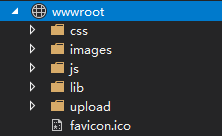
在我们的Web程序开发中,肯定要提供很多的静态文件(比如:JS,CSS)给客户端下载使用.所以我们先来看看ASP.NET Core中是怎么处理的.

当我们创建一个ASP.NET Core MVC的模版程序后,会发现它与传统的文件结构还是会有区别.

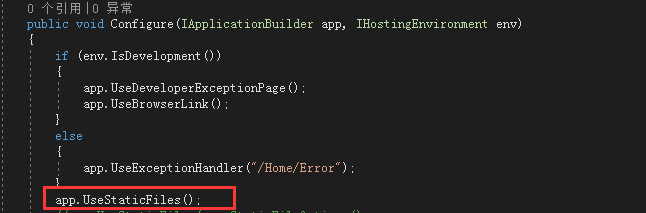
多了一个wwwroot文件夹,少了很多其他的资源文件夹.如图:

我们点进去,就可以看到,微软的模版,已经把所有的资源文件全放到了这里面,如图:

这样,项目就干净了许多,下面,我们就来讲讲这个资源根目录

## 1.提供静态文件

我们到模版的Startup管道配置中,可以看到,注入相关静态资源的代码,已经帮我们写好了,如图:



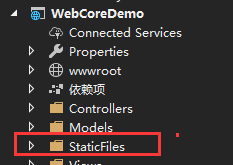
这句话就是注入静态资源用的,默认会将wwwroot的资源直接配置的和根目录一样,来方便访问.

访问资源的URL类似: "http://localhost:9189/images/banner3.svg" 这样.

那么问题来了,我们能不能自己配置这个静态资源呢?.

当然是可以的~.

我们在项目中创建文件夹如下:



然后通过**StaticFileOptions**注入静态资源的配置,代码如下:

[IMG_260](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

app.UseStaticFiles(new StaticFileOptions()

{

FileProvider = new PhysicalFileProvider(

Path.Combine(Directory.GetCurrentDirectory(), @"StaticFiles")),

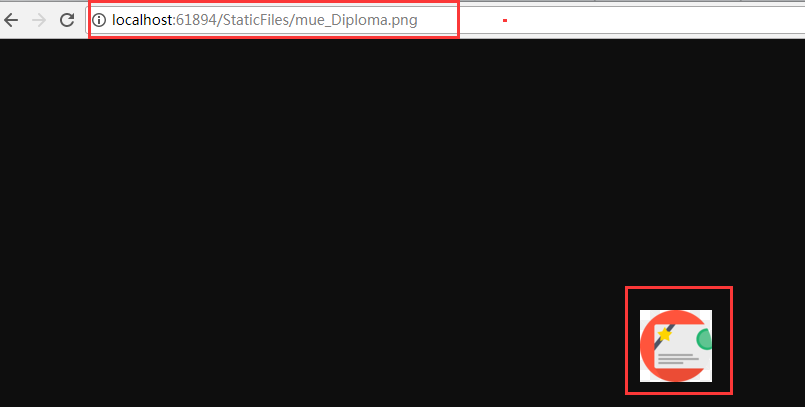
RequestPath = new PathString("/StaticFiles")

});

[IMG_261](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

**注意:这里的2个StaticFiles, 第一个是你的本地资源路径,第二个是你需要配置的URL路径,URL路径可以自己定义,这样可以一定程度上保护自己的资源安全.**

然后我们通过URL访问效果如下:



在StaticFileOptions中,我们还可以通过OnPrepareResponse属性配置我们的响应头,添加 代码如下:

[IMG_263](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

app.UseStaticFiles(new StaticFileOptions()

{

FileProvider = new PhysicalFileProvider(

Path.Combine(Directory.GetCurrentDirectory(), @"StaticFiles")),

RequestPath = new PathString("/StaticFiles"),

OnPrepareResponse = ctx =>

{

ctx.Context.Response.Headers.Append("Cache-Control", "public,max-age=600");

}

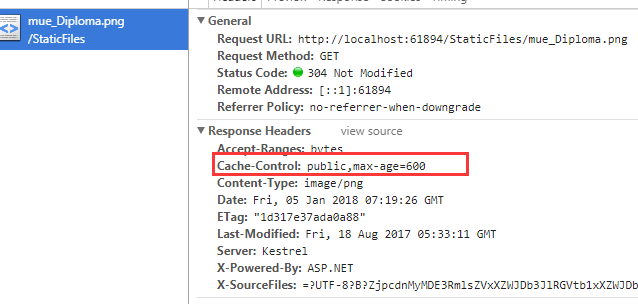
});

[IMG_264](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

 这里,我们设置http响应缓存为600秒.

(小知识:max-age：表示当访问此网页后的max-age秒内再次访问不会去服务器请求，其功能与Expires类似，只是Expires是根据某个特定日期值做比较。一但缓存者自身的时间不准确.则结果可能就是错误的，而max-age,显然无此问题.。Max-age的优先级也是高于Expires的。)

效果如下:



这样,我们就可以根据需求来配置自己的静态文件缓存和其他的响应头信息.

## 2.启用静态文件目录浏览

开启静态文件目录浏览..其实是一件安全性级低的事情,不管是传统的asp.net还是asp.net core 默认都是关闭了这个功能.

但是,不排除我们会用到.在asp.net中,我们只需要的Web.config中配置即可.

下面我们就来讲讲如何在asp.net core 中启用我们的静态文件目录游览

开启静态文件目录游览需要使用UseDirectoryBrowser来注入配置,代码如下:

[IMG_266](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

app.UseDirectoryBrowser(new DirectoryBrowserOptions()

{

FileProvider = new PhysicalFileProvider(

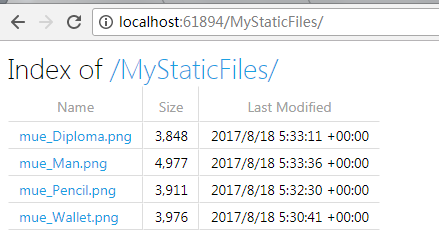
Path.Combine(Directory.GetCurrentDirectory(), @"StaticFiles")),

RequestPath = new PathString("/MyStaticFiles")

});

[IMG_267](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

这里,我特意改了URL的访问路径.为MyStaticFiles..我们来看看效果:

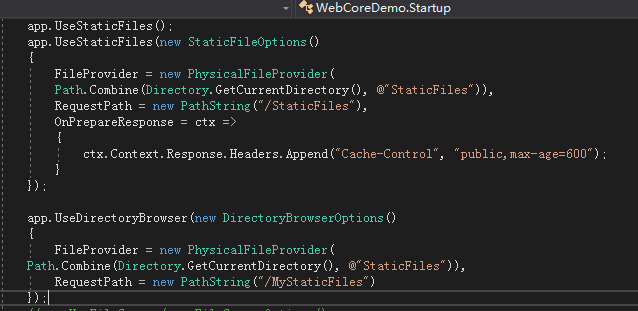


我们点击里面任意一个文件,会发现.返回404.因为路径是

所以,这里需要和上面的文件URL路径一致,才可以访问的到文件.**(这样其实提供了一种比较安全的配置手段.可以隐藏自己本机的真实路径)**

## 3.使用UseFileServer合成的注入方法~简化代码.

按照我们上面的配置,如果开启了文件和目录游览,就会发现写了不少注入代码,如下:



这样不是很方便,也不利于后期的维护.那么,下面我们就来简化他~

UseFileServer的**功能结合了**UseStaticFiles**，**UseDefaultFiles**和**UseDirectoryBrowser**。**

我们把上面的代码全部注释.并修改代码如下:

[IMG_271](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

app.UseFileServer(new FileServerOptions()

{

FileProvider = new PhysicalFileProvider(

Path.Combine(Directory.GetCurrentDirectory(), @"StaticFiles")),

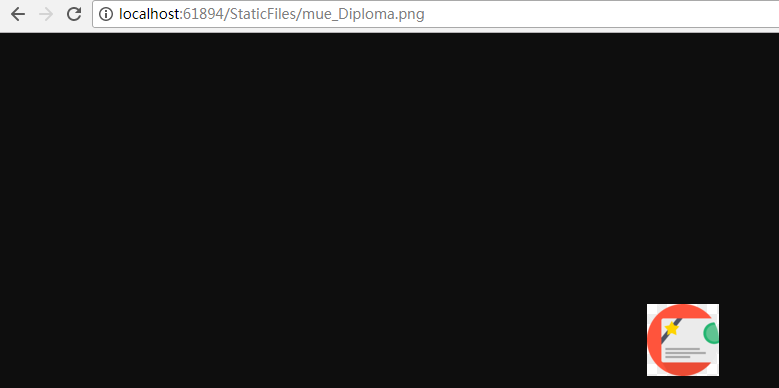
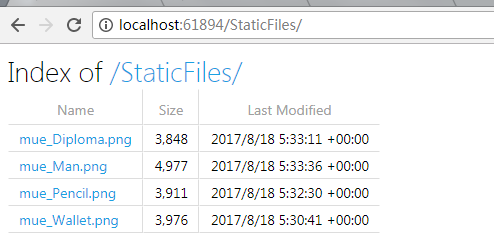
RequestPath = new PathString("/StaticFiles"),

EnableDirectoryBrowsing = true

});

[IMG_272](https://www.cnblogs.com/GuZhenYin/p/javascript:void(0);)

效果如下:



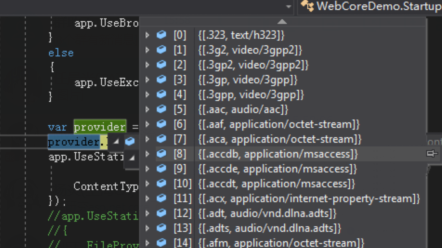
**这里,注意EnableDirectoryBrowsing属性,就是是否启用目录的属性.**

## 4.ASP.NET Core中使用MIME内容类型来过滤静态文件返回.

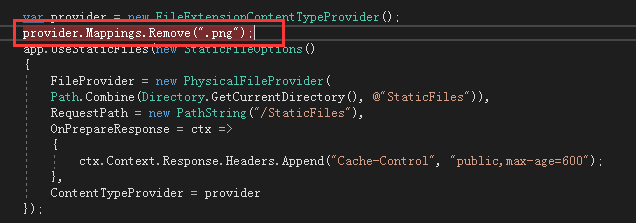
上篇文章我们提到过.需要使用

## FileExtensionContentTypeProvider

这个类.具体使用方法,直接new一个即可,里面包含了300多种已知的文件类型,如下:



为什么这里我们要提到这个呢.因为他可以帮我们过滤一些不想让客户端访问的文件类型.如下:



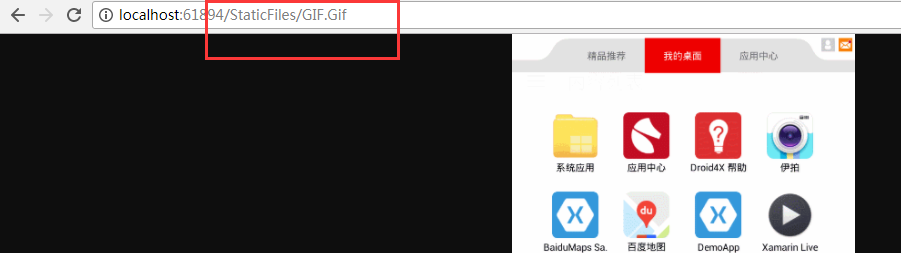
**这里要注意,使用FileExtensionContentTypeProvider,只能通过UseStaticFiles,不能使用UseFileServer**

**因为ContentTypeProvider是StaticFileOptions(静态文件选项)的配置内容,在FileServerOptions中并没有.**

上面的代码,我们删除了png格式文件.也就是不返回这个内容的文件.运行后效果如下:



我们访问我们的gif格式的文件,效果如下:



还是可以正常访问的.对于一些安全性较高,或者涉及文件类型较多的功能来说.还是比较实用的**.可以自己过滤掉一些敏感类型.比如exe.**